Title
A solution to the crucial problem of population degeneration in high-dimensional evolutionary optimization

Journal
IEEE Systems Journal, 5(3)

ISSN
1932-8184

Authors
Chu, W
Gao, X
Sorooshian, S

Publication Date
2011-09-01

DOI
10.1109/JSYST.2011.2158682

A Solution to the Crucial Problem of Population Degeneration in High-Dimensional Evolutionary Optimization

Wei Chu, Xiaogang Gao, and Soroosh Sorooshian

***Abstract*—Three popular evolutionary optimization algorithms are tested on high-dimensional benchmark functions. An important phenomenon responsible for many failures – "population degeneration" – is discovered. That is, through evolution, the population of searching particles degenerates into a subspace of the search space, and the global optimum is exclusive from the subspace. Subsequently, the search will tend to be confined to this subspace and eventually miss the global optimum. Principal components analysis (PCA) is introduced to discover population degeneration and to remedy its adverse effects. The experiment results reveal that an algorithm's efficacy and efficiency are closely related to the population degeneration phenomenon. Guidelines for improving evolutionary algorithms for high-dimensional global optimization are addressed. An application to highly nonlinear hydrological models demonstrates the efficacy of improved evolutionary algorithms in solving complex practical problems.**

***Index Terms*—Differential evolution, evolutionary computation, high-dimensional, particle swarm optimizer, principal components analysis, shuffled complex evolution (SCE-UA).**

I. Introduction

EVOLUTIONARY optimization algorithms for unconstrained optimization attempt to discover the value and location for the global optimum of an m-dimensional function:

$$f(\hat{g}) = \min_{x \in R^m} f(x) \text{ or } \hat{g} = Arg\min f(x), \text{ where } f : R^m \rightarrow R.$$

In the search, the solution vector $\hat{g} \in R^m$ is solely derived from the values of the function without the use of the function's derivatives. The objective function can be in the form of either a mathematical expression or a physical model's computer program, which makes the optimization algorithms a variety of applications in science, engineering, and business [1]–[10]. Current evolutionary algorithms are mostly featured with particle-based evolution processes: the algorithm first randomly distributes a "population" of "particles" (or members, points) $\{x_i\}_{i=1}^{NS}$,with $x_i \in R^m$ and NS is the size of population (number of samples), to sample the objective function $\{f(x_i)\}_{i=1}^{NS}$ in the search space, then sequentially evolve the population by substituting the existing particles (as parents) with new ones having better function values (as offspring). After many generations of evolution, the population is expected (but not guaranteed) to have at least a particle reaching the global optimum.

Recently, a large number of evolutionary algorithms or algorithm modifications have been developed using genetic-, swarm-, annealing-, and hybrid-based mechanisms to enhance the reliability and efficiency of the evolution processes. These algorithms supported by powerful computation capability have shown good performances in global optimization and become popular increasingly. However, even with these algorithms, global optimization can still be problematic or computationally demanding for high-dimensional cases ($\text{dimensionality} \geq 30$). High-dimensional problems challenge evolutionary algorithms because the feasible space grows exponentially with increase in dimensionality and the impracticality of increasing the population size in the same manner. Many existing evolutionary algorithms do not pay enough attention to this aspect and sacrifice efficacy due to implementing searches of high computational demand. Therefore, understanding the mechanisms of algorithms that cause success (or failure) and efficiency (or inefficiency) of searching in high-dimensional spaces, is a prerequisite to overcome the theoretical barriers. In this study, we use high-dimensional benchmark functions to test three popular evolutionary algorithms. The selected algorithms are the Shuffled Complex Evolution (SCE-UA)—an algorithm based on simplex scheme, the Particle Swarm Optimizer (PSO)—an algorithm simulating the social activity of animal groups; and the Deferential Evolution (DE)—an algorithm using genetic hybridization.

Although the search strategies of each of these algorithms originate from distinct scientific fields, we show that they all possess two opposite functionalities: 1) exploitation: the process of making particles converge to the global optimum, and 2) exploration: the process to enable particles to explore the feasible space of parameters. The exploitation process tends to drive particles to the most prominent region in the feasible space and speed up the search. Meanwhile, this process increases the risk of missing the global optimum. The exploration process attempts to overcome this problem by diversifying the search directions. This process increases robustness of the search, but at the expense of dragging down the speed of convergence. Detailed analysis of the selected algorithms' processes and performances on high-dimensional benchmark functions revealed that the barriers which keep each algorithm from success of efficient optimization on high-dimensional problems are closely related

Manuscript received February 28, 2011; revised May 10, 2011; accepted May 10, 2011. This work was supported in part by the UCOP program of the University of California (Grant 09-LR-09-116849-SORS), and by the ROSES program of NASA (Grants NNX09AO67G and NNX NNX06AF93G).

The authors are with the Department of Civil & Environmental Engineering, University of California, Irvine, CA 92697-2175 USA (e-mail: wchu2@ uci. edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSYST.2011.2158682

TABLE I
PSEUDO CODES FOR STUDIED ALGORITHMS

(1) Randomly initiate a population of NS particles $\{x_i\}_{i=1}^{NS}$ within the search domain in R^m

$\{x_{ip} = U[b_p, a_p]\}_{p=1}^{m}$, i: particle index; p: parameter index; a_p, b_p: the upper and lower bounds for parameter p.

(2) Calculate the function values of individual particles: $\{f(x_i)\}_{i=1}^{NS}$,

Record the population's minimum function value and position: $f_{best}, \hat{g}$,

Set the maximum number of function evaluation: I_{max}.

SCE	*PSO*	*DE*
Iteration = 1 3)Order particles by their function values: $f(x_1) \le f(x_2) \le \ldots \le f(x_{NS})$ Shuffle ranked particles $\{x_i\}_{i=1}^{NS}$, into n complexes C_k, each has $(2m+1)$ particles Note: $NS = n\,(2m+1)$ 4) Evolve each complex via $2m+1$ simplex subroutines. For $i = 1,\ldots, n$ complex loop For $j = 1, \ldots, 2m+1$ simplex loop Select $m+1$ particles to form a simplex Run the Nelder-Mead simplex scheme End End if function evaluation < I_{max} and population geometric size is greater than the threshold goto (3) end if Stop	Record each particle's best position $\hat{x}_i$ Iteration = 1 (3) Evolve via each particle: For $i = 1,\ldots, NS$ Generate two random vectors $r_1, r_2 \in R^m$, $r_{1p}, r_{2p} = U[0,1]$ Set a displacement vector v_i $v_i = c_0 v_i' + c_1 r_1 \otimes (\hat{x}_i - x_i)$ $+ c_2 r_2 \otimes (\hat{g} - x_i)$ candidate: $\tilde{x}_i = x_i + v_i$ if $f(\tilde{x}_i) \le f(x_i), x_i = \tilde{x}_i$, end if if $f(\tilde{x}_i) \le f(\hat{x}_i), \hat{x}_i = \tilde{x}_i$, end if if $f(\tilde{x}_i) \le F(\hat{g}), \hat{g} = \tilde{x}_i$, end if End Iteration = Iteration +1 if function evaluation < I_{max} goto (3) end if Stop	Iteration = 1 (3) Evolve via each particle: For $i = 1,\ldots, NS$ Randomly select 3 particles $x_a, x_b, x_c \in \{x_j\}_{j=1}^{NS};\ a,b,c \ne i$ Construct a new point v_i $v_i = x_a + F(x_b - x_c)$ Hybridize components of x_i, v_i $x_i = (x_{i1},\ldots, x_{im})$ $v_i = (v_{i1},\ldots, v_{im})$ For $p = 1,\ldots, m$ $r = U[0,1]$ $\tilde{x}_{ip} = \begin{cases} v_{ip} & \text{if } r \le C \\ x_{ip} & \text{if } r > C \end{cases}$ End if $f(\tilde{x}_i) < f(x_i),\ x_i = \tilde{x}_i$, end if if $f(\tilde{x}_i) < f(\hat{g}),\ \hat{g} = \tilde{x}_i$, end if End Iteration = Iteration +1 if function evaluation < I_{max} goto (3) end if Stop

to implementations of these two functionalities. Balancing these two functionalities is a key aspect of constructing a successful algorithm.

In this paper, we report the results of our study which reveals that in high-dimensional searches, some exploitation processes tend to drive particles into a subspace of the feasible space. In other words, at certain stages during the evolution, all the particles in the population move into a subspace, which has smaller dimensionality than that of the search space. Subsequently, the search will tend to be confined to the subspace and eventually fail if the global optimum is excluded from the subspace. We refer to this phenomenon as "population degeneration." Principal components analysis (PCA) [11] of particle population is used to detect the occurrence of degeneration. Furthermore, principal components (PCs) can provide information useful to remedy the adverse effects of population degeneration.

II. SELECTED ALGORITHMS

Three among the most popular evolutionary optimization algorithms are investigated in this study. The pseudo codes for the selected algorithms are summarized in Table I. The algorithms all start from an initial population (first-generation) of randomly sampled particles within the search range in the feasible space. Afterwards, the offspring replacements with different mechanisms are initiated in order to evolve the population.

A. Shuffled Complex Evolution (SCE-UA)

The SCE-UA algorithm [12], [13] employs the Nelder–Mead simplex scheme [14], [15] to make particle replacements for population evolution. An m-simplex is an m-dimensional "pyramid" (convex hull) in R^m with $m+1$ affinely independent vertices $\{x_i \in R^m\}_{i=1}^{m+1}$ and C_{m+1}^2 edges $\{x_i - x_j\}_{i,j=1;i\ne j}^{m+1}$(e.g., 1-D simplex is a line segment; 2-D simplex is a triangle; 3-D simplex is a tetrahedron.). Starting with a simplex located inside the search domain, the scheme first ranks the vertices according to their function values, so that $f(x_1) \le f(x_2) \le \cdots \le f(x_{m+1})$, then it attempts to replace the worst vertex x_{m+1} with a new point $x(\delta) = (1+\delta)\overline{x} - \delta x_{m+1}$ on the line between x_{m+1} and the centroid of the rest vertices, $\overline{x} = (1/m)\sum_{i=1}^{m} x_i$. Parameter δ determines the location of the new point on the line and several δ values (in SCE-UA, $\delta = 1$ for reflection, and $\delta = -0.5$ for contraction) will be tried to find a new point having $f(x(\delta)) \le f(x_{m+1})$. If successful, the worst vertex is replaced by the new point, $x_{m+1} = x(\delta)$ otherwise, x_{m+1} is replaced by a random point (mutation) in the search domain. In this scheme, information from a simplex's vertices on the response surface of the benchmark function is used to approximate the direction for steepest descent. Driven by the direction of steepest descent, the simplex can effectively find a

better offspring $x(\delta)$. To prevent simplexes from converging to local optima and to enhance the chances of finding the global optimum, SCE-UA utilizes the shuffled-complex process. At the start, the particle population is partitioned into n complexes. Each complex includes $2m+1$ particles and evolves independently using the simplex method. At the beginning of every iteration of evolution, a simplex is formed within each complex by randomly selected $m+1$ out of the $2m+1$ particles in order to perform simplex search. Once a worst vertex is replaced, the simplex will be broken down and its particles return to the complex. We refer to this procedure as simplex subroutine. After each complex completes a certain number ($2m+1$ in this study) of simplex subroutine, particles in all complexes are mixed, sorted and re-divided into new complexes through the shuffling procedure (see Table I), and one iteration ends. By shuffling, the new complex is likely to contain particles from all the previous complexes, hence having information about the function over the region covered by the entire population instead of a single complex. This gathering of information from all the previous complexes results in a better chance for the new complexes moving towards the global optimum.

B. Particle Swarm Optimizer (PSO)

The concept of PSO [16], [17] originates from the computer simulation of the social activities of a bird flock or fish school in which individual members can benefit from its own experience and other members' best discovery during their search for food, mates, or better living conditions. The algorithm starts from a randomly selected initial population and evolves individual particles successively. Each particle's position x_i is updated by trying a displacement (velocity) based on three sources: (1) the particle's velocity in the previous evolution ($v_i{}'$), (2) the particle's best ever position ($\hat{x}_i$), and (3) the population's current best position ($\hat{g}$):

$$v_i = c_0 v_i{}' + c_1 r_1 \otimes (\hat{x}_i - x_i) + c_2 r_2 \otimes (\hat{g} - x_i)$$
$$\widetilde{x}_i = x_i + v_i$$

where c_0, c_1, and c_2 are the significance coefficients and $r_1, r_2 \in R^m$ are random vectors with uniformly generated components $\{r_p = U(0,1)\}_{p=1}^m$. $\vec{a} \otimes \vec{b} = \vec{c}$ is defined as producing a new vector having components $(c_p)_{p=1}^m = (a_p b_p)_{p=1}^m$. If $f(\widetilde{x}_i) \leq f(x_i), x_i = \widetilde{x}_i$, otherwise no replacement occurs. A particle's velocity is bounded by the maximum velocity $V_{\max}$.

In the algorithm, taking the displacement with reference to the best point in the population makes a particle have better chance to find an offspring and drives particles moving towards a convergence. In contrast, the particle movements will diverge strongly by historical and current information ($\hat{x}_i$ and v_i') from individual particle's path and the random vectors that alter the component magnitudes of the displacements.

C. Differential Evolution (DE)

The DE algorithm [18] uses the so-called "greedy" scheme to generate offspring. Similar to the PSO algorithm, it attempts to replace the population's particles one-by-one with an offspring. In the DE algorithm, for a particle x_i, the candidate offspring $\widetilde{x}_i$ has hybridized components $\{\widetilde{x}_{ip}\}_{p=1}^m$ from x_i and another random variable v_i according to a randomly generated number $r = U[0,1]$:

$$v_i = x_a + F(x_b - x_c),\ (0 < F \leq 2)$$
$$x_a, x_b, x_c \in \{x_j\}_{j=1}^{NS}\ (a, b, c \neq i)$$
$$\widetilde{x}_i = \left(\widetilde{x}_{ip} = \begin{cases} v_{ip}, & \text{if } r \leq C \\ x_{ip}, & \text{if } r > C \end{cases} \right)_{p=1}^m$$

where $C < 1$ is the crossover constant, F is the scaling factor and p is the component index. If $f(\widetilde{x}_i) \leq f(x_i)$, $x_i = \widetilde{x}_i$, otherwise the parent survives to the next evolution.

In DE algorithm, the use of random vector v_i is a loose process that tends to have the offspring generated along random directions. The hybridization process also helps making population evolve randomly.

SCE-UA, PSO, and DE are three very typical methods and posse distinct search strategies. The SCE-UA algorithm is a steepest descent method, employing $m+1$ particles (i.e., the simplex vertices) to approximate slope of the objective function's response surface in order to speed the evolution. On the other hand, the DE algorithm is an method heavily relying on randomness such that the discovery of an offspring relies on a randomly selected set of three particles from the population. As for PSO, its search strategy falls somewhere between SCE-UA and DE. In SCE-UA, the divergence process (the shuffled-complex procedure) is at the particle level while in both PSO and DE algorithms, the particles are broken into components for more diverse searches. Those differences in search strategy significantly affect their global optimization performances on high-dimensional problems, as demonstrated through the benchmark function tests

ALGORITHM EFFICIENCY AND ROBUSTNESS

D. Benchmark Functions

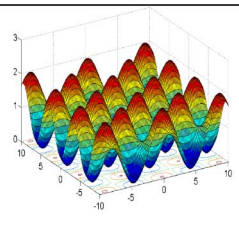
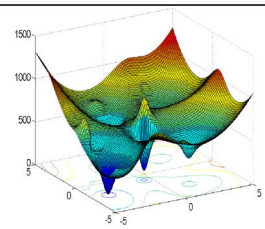
We chose two benchmark test functions that represent two major types of test functions: classical and compositional. The first one is the popular Griewank function. Like many classical benchmark functions, Griewank function has a regular and symmetrical shape. As shown in Table II, this function is constructed with two components: 1) an m-D sphere function $f_1 = \sum_{p=1}^m (x_p^2/4000)$ that gives the global minimum at the origin, and 2) an m-D wave function $f_2 = 1 - \prod_{p=1}^m \cos(x_p/\sqrt{p})$ that overlies on the trend function, f_1, and creates many local minima surrounding the global minimum (see the 2D response surface in Table II). This feature makes the Griewank function serve well for testing an algorithm's capability of escaping local minimums and converging to the global minimum.

The second benchmark function is a composition function (CF1) suggested by Liang *et al.* [19]. As shown in Table II, the composition function is a weighted summation of ten sphere functions:

$$\underset{x \in R^m}{f(x)} = \sum_{i=1}^{10} w_i(x) \left[f_i \left(\frac{(x - o_i)}{0.05} \right) + bias_i \right].$$

By changing vector o_j and scalar $bias_j$ the sphere functions can have variable magnitudes and can be located at different positions in the feasible space. Weight functions are used to em-

TABLE II
BENCHMARK FUNCTIONS

Griewank's function	Composition Sphere function
$f(x)=\sum_{p=1}^{m}\frac{x_p^2}{4000}-\prod_{p=1}^{m}\cos(\frac{x_p}{\sqrt{p}})+1$	$f(x)=\sum_{i=1}^{n}w_i(x)[f_i'((x-o_i)/0.05)+bias_i]$, $i=1,\ldots,10$ n sphere functions $f_i(x)=\sum_{p=1}^{m}x_p^2$, $x=\{x_p\}_{p=1}^{m}\in R^m$ $f_i'(x)=2000f_i(x)/\lvert f_{\max i}\rvert$ $f_{\max i}=\max\{f_i(x),i=1,\ldots,10\}$ $w_i(x)=\exp(-\frac{f_i(x-o_i)}{2m})$, $w_i=\begin{cases}w_i & \text{if } w_i=\max(w_i)\\ w_i(1-\max(w_i)^{10}) & \text{if } w_i\neq\max(w_i)\end{cases}$ $w_i=w_i/\sum_{j=1}^{n}w_j$
	
Search Range: $[-10, 10]^m$	Search Range: $[-5, 5]^m$

m is dimensionality, and plots illustrate function response surface when $m = 2$.

phasize the impact of the corresponding sphere function in the region around its sphere center. Unlike most traditional benchmark functions, the CF function's response surface is characterized with multiple irregular attractive regions (with distinct shapes and sizes) converging to individual minimums at different levels. The lowest one is the global minimum. This irregularity poses a great challenge for evolutionary algorithms, especially when the global minimum is located within a small convergence zone close to the border.

Noticeably, both the benchmark functions are continuous and smooth (differentiable everywhere). The points where the function reaches either local or the global minimums have zero gradient ($\nabla f = 0$). These points are identified as the stationary points.

E. Experimental Setup

The SCE-UA, PSO and DE algorithms are compared on both benchmark functions of 30-D and 100-D. The maximum number of function evaluation is set at 500 000 for the 30-D experiments and 1 000 000 for the 100-D experiments, respectively. Each of the experiments is run 30 times with different random seeds. All algorithms' runs start with an initial population of $(4m + 2)$ particles, where m is the dimensionality of the benchmark function.

Parameters in each algorithm are specified to their default or popular values:

1) SCE-UA: The number of complex (n): $n = 2$. Each complex has $2m+1$ particle. Therefore, the population size is $4m+2$ particles.

2) PSO: The significant coefficients: the inertia weight is held as a constant $c_0 = 0.5$. (Based on preliminary experiments, we did not found noticeable difference in performance between the use of an annealed and a constant c_0. This is in accord with the study of [20]) c_1 and c_2 are set at 2. $V_{\max}$ is half of the search range.

3) DE: The crossover constant $C = 0.5$; the scaling factor $F = 0.5$.

As reported in [21], bound handling is critical to the performance of PSO. The widely used random and absorbing bound handling schemes may paralyze PSO when applied to high-dimensional problems. The difference in effects from different bound handling on SCE-UA and DE are not as evident as on PSO. Therefore, in this study, we adopt reflecting bound handling method for all three algorithms. In this method, when a particle flies outside a bound in one of the dimensions, the bound will act like a mirror and reflect the projection of the particle's displacement.

F. Results

1) Results for the Griewank Function: As illustrated in Fig. 1(a) and (b), the three algorithms perform consistently through their 2×30 runs. In every SCE-UA and DE run, the best function value (BFV) converges to the global minimum. For PSO, a few runs converge to local minimums (eight runs in 30-D experiment and five runs in 100-D). SCE-UA converges at least one-order faster than PSO and DE in both 30-D and 100-D cases. On the other hand, PSO outperforms DE in terms of convergence speed. With the increase in dimensionality of the benchmark function, differences in BFV decreasing rates of two algorithms are more evident in 100-D than in 30-D. In particular, for SCE-UA, the numbers of function evaluations to convergence remains at an order of 10^3 in both 30-D and 100-D runs, whereas the numbers for PSO and DE go up from the order of 10^4 to 10^5. These results indicate that, in the case of Griewank function, the increase in dimensionality causes the efficiencies of PSO and DE to drop more significantly than that of SCE-UA.

2) Results for the CF1 Function: In Fig. 1(c) (the 30-D experiments), for SCE-UA, all the runs reach the close vicinity of global minimum, but are unable to converge to the global minimum. As geometric sizes of populations become very small, all of the optimization runs terminate prematurely as though they had converged to a minimum point. These termination points randomly distributes around $\hat{g}$ with mean BFV of 1.3295 and mean distance to $\hat{g}$ of 0.1943. Most PSO runs succeed while only 6 runs converge to one of the local minima and are trapped there within the given maximum number of function evaluations. The DE runs unanimously achieve the global minimum. Similar to the results on the Griewank function, SCE-UA shows the fastest BFV decrease rate; BFVs in the successful PSO runs drop faster than those in DE runs, where as in the failed PSO runs, BFV drops even slower than in DE runs.

The results from the 100-D experiments (Fig. 1(d)) show the problems more clearly. All the SCE-UA runs quickly stop at some points that are still away from the global minimum. The majority of PSO runs succeed in converging to the solution with nine runs are trapped to a local minimum. The DE algorithm exhibit its robustness and discover the global minimum without

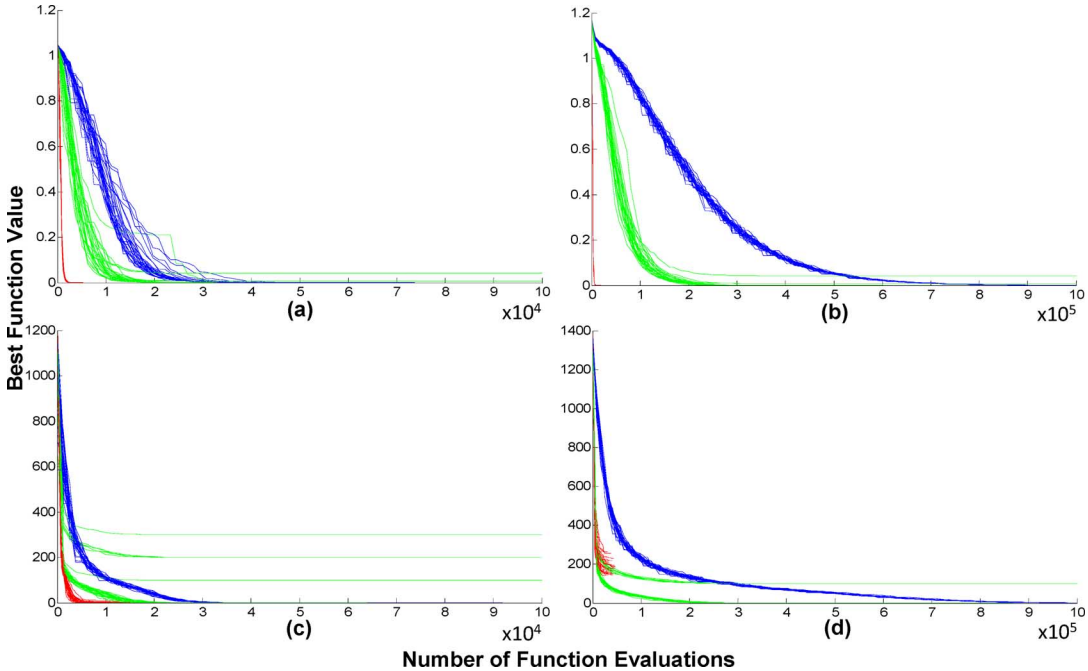


Fig. 1. Fitness curves for SCE-UA (red), PSO (green) and DE (blue) on two benchmark functions and in 30-D and 100-D experiments (All 30 runs are plotted.) for a) 30-D Griewank function, b) 100-D Griewank function, c) 30-D CF1 function, and d) 100-D CF1 function.

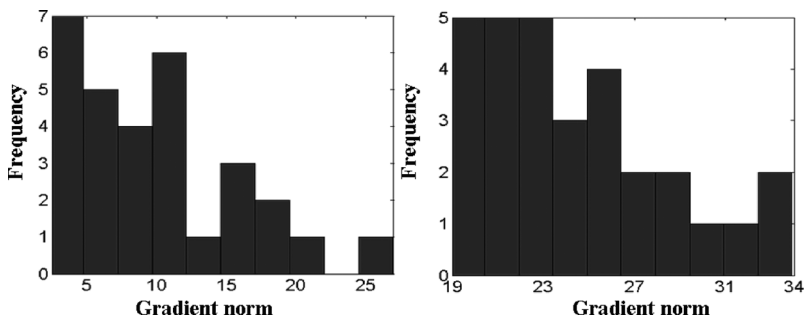


Fig. 2. Histograms of gradient norms at final points of SCE-UA runs on 30-D (left) and 100-D (right) CF functions. (Gradients are numerically estimated with $\Delta x = 1 \times 10^{-6}$).

exception. The BFV decreasing speeds of individual algorithms express the similar patterns as in the 30-D experiments.

To understand the algorithms' different behaviors, we investigated the results further. The questions posed were as follows.

- Where do all the SCE-UA runs stop at?
- What impede the SCE-UA algorithm from converging to the global minimum?
- Why DE is so robust that it succeed in every scenario?

III. Principal Components Analysis and Population Degeneration

A. Stagnation of the SCE-UA Algorithm

As described earlier, the SCE-UA algorithm employs the Nelder–Mead simplex scheme to move the offspring along the function's steepest descent direction. Ideally, if the function is continuous and smooth, the scheme will eventually drive the simplex's vertices converge to a stationary point. However, it has been reported that even for lower-dimensional problems, Nelder–Mead simplexes can sometimes stagnate at non-stationary points, a phenomenon named "stagnation" [22]. Our tests using the CF1 function reveal that incidents of stagnation increase when the function dimensionality increases. To illustrate where the SCE-UA runs stop, we examine the function gradients (numerically estimated with $\Delta x = 1 \times 10^{-6}$) at the final points of SCE-UA runs in both 30-D and 100-D cases. We find that all the final points are non-stationary ($\nabla f \neq 0$). Fig. 2 shows the histograms of gradient norms at the final points: all the points possess significant gradients.

To identify if these final points are located inside the attractive region of the global minimum, we further examine the geometric relationship between the normalized negative gradient and the normalized direction vector from the final point to the global minimum for each final point. Given the fact that the CF1 function over the attractive region of the global minimum is a weighted sphere function, at any point in this region the negative gradient vector should point towards the global minimum. As illustrated in Fig. 3, for 30-D runs, the corresponding components of these two normalized vectors at all the final points are highly correlated. This indicates that these points are already close to the global minimum with negative gradients pointing towards it. For 100-D runs, the final points are still located inside the attractive region but the correlations are not as high as in the 30-D

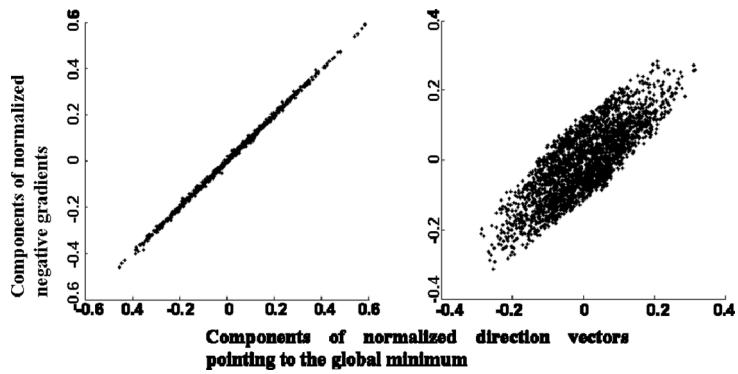


Fig. 3. Correlation between components of normalized negative gradient and normalized direction vectors pointing to the global minimum at final points for SCE-UA runs on the 30-D (left) and 100-D (right) CF functions.

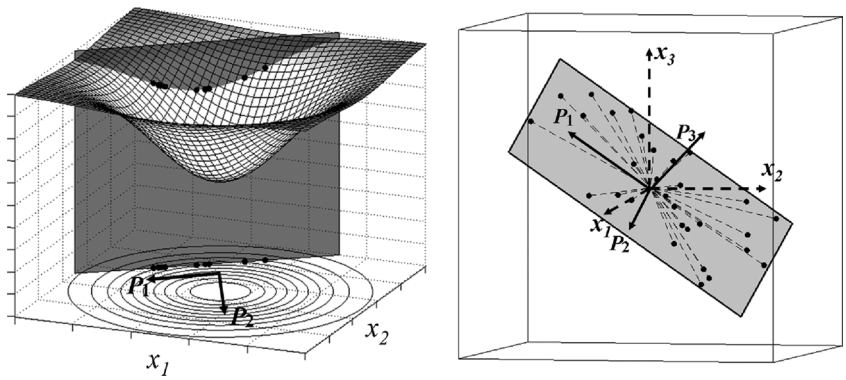


Fig. 4. Illustration of population dimensionality reduction in 2-D (left) and 3-D (right) scenarios. In 2-D, the benchmark function response surface is shown as well as the contours in the coordinate plane. When the population degenerates onto a line parallel to the first PC (P1) with the variance on the second PC (P2) equal to zero, the succeeding search will be restricted on the line and eventually converges to the best point on this line which is a non-stationary point in the original feasible space. Similarly, in 3-D all sample particles may degenerate onto a plane defined by the first two PCs (P1 and P2) with the third orthogonal PC (P3) perpendicular to the plane. If the global minimum $\hat{g}$ is not on this plane, the search will lose the capability of converging to $\hat{g}$. (Black dots are searching particles.).

cases because these final points are relatively farther away from the global minimum.

A key question here is: what make the SCE-UA algorithm lose its capability to fully converge? A possible reason is that the particles which are randomly selected from a complex to form a simplex have already lost some properties necessary for achieving the global optimum. We hypothesized that after generations of evolution the population might have degenerated to a subspace of the feasible space. In order to test this hypothesis, we need to examine the geometric structure of the population in a way which illustrates the faults of the particle collection. For this, we used the principal components analysis.

B. Principal Components Analysis (PCA)

Principal Components Analysis (PCA) is a multivariate analysis tool that transforms a given dataset to a new orthogonal independent coordinate system so that the first coordinate (called the first principal component PC) has the largest variance of projections from the dataset; the second coordinate has the second largest variance and so on. In certain cases, some lower-rank PCs will have negligible variances, which means the dataset having "dimensionality reduction". In our case, the population's particles $\{x_i \in R^m\}_{i=1}^{NS}$ are transformed to the PC coordinate system $\{y_i \in R^m\}_{i=1}^{NS}$:

$$y_i = Ax_i$$
$$\lambda_k P_k = BP_k$$

where matrix A has PCs ($P_k : k = 1, \ldots, m$) as its columns, B is the covariance matrix of the particle population in the original coordinate system, and eigenvalue λ_k is the data variance along P_k.

If we have $\lambda_{\mathrm{L}}/\sum_{j=1}^{m} \lambda_j \to 0$, the vector component $y_{ip} \to$ constant for every $p \in \{L \le p \le m\}$. This indicates that the dataset is located within a subspace spanned by $(L-1)$ independent orthogonal vectors (PCs) denoted as R^{L-1}.

By applying the PCA procedure to a population and checking the PC's relative variance we can identify if the dimensionality reduction has happened in the population. If it has happened, we need to be aware of two issues:

1) Given the global optimum location of the benchmark function $\hat{g}$, it is easy to detect if $\hat{g}$ is inside the subspace. If $\hat{g}$ is outside of the subspace, the particle population may lose the ability to converge to $\hat{g}$ or may even converge to a non-stationary point. Sketchy plots in Fig. 4 provide visualization of reduced dimensionality of particle population in 2-D and 3-D scenarios and the consequences.
2) If vectors $y_{i_1}, y_{i_2} \in R^{L-1}$ and $y_k = c_1 y_{i_1} - c_2 y_{i_2}$, then $y_k \in R^{L-1}$ too. It means that theoretically, offspring generation mechanism based on linear operations on parents in degenerated population cannot restore the lost dimensions.

In high-dimensional optimization problems, one can not ensure that particle population is able to maintain it dimensionality throughout the whole search. Actually, we discovered that in each of the SCE-UA failed runs and most of PSO problematic

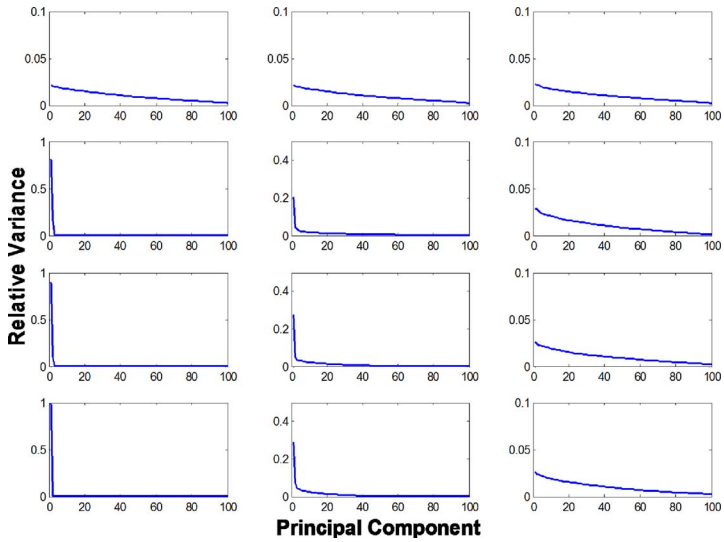


Fig. 5. The PCA results during the evolution of the SCE-UA (left column), PSO (middle column) and DE (right column) runs for the 100-D CF1 function. The horizontal coordinate represents the 100 PCs ordered according to their relative variances (the vertical axis). From top to bottom, the row panels are for the results after the first iteration, after one-third of the total iterations, after two-thirds of the total iterations, and after the last iteration, respectively.

runs, the population degenerated: all particles spanned a subspace with a reduced dimensionality and the benchmark function's global minimum point was excluded from the subspace. Detailed results are illustrated in the following section.

C. PCA Results of the Three Algorithms

In the experiments, we applied the PCA procedure to the particle populations every time a generation of evolution (iteration) is completed. Due to the characteristics of the PSO algorithm, the best historical position of each particle is also included in the PSO population. Fig. 5 illustrates how the relative variances on PCs varies from the early population to the final one for median runs of three algorithms on the 100-D CF1 function (results from other 100-D runs and the 30-D runs are similar). Since all algorithms start with a randomly sampled initial population, after the first iteration, the populations of all three algorithms still maintain comparable variances for each individual PC (the top panels). However, with continuing evolution, the PCA results for the three algorithms diverge.

For SCE-UA (the panels on the left column), the population finally generates so much that only one PC has dominant amount of variance, 99.5% of the total, after 74 iterations (about 4.5×10^4 function evaluations). In fact, after one third of the total iterations, only four PCs have significant variances. This means that the population is confined in a 4-D subspace of the 100-D search space. Because the SCE-UA algorithm generates the candidate offspring along the steepest descent direction – a difference vector between a simplex's worst vertex and the centroid of the rest vertices, if all the vertex particles are within the same subspace, the new offspring will be confined in the same subspace. The population continues reducing its dimensionality until quits at a 1-D subspace and the final point is not even a stationary point.

For PSO (the panels in the middle column), the degeneration of particle population is less severe than in SCE-UA. After two-third of the total iterations (2120 iterations with about 8.6×10^5 function evaluations), the last 28 PCs out of all 100 PCs have only 1% of the total variance, which means that the capability of searching over directions of these 28 PCs is greatly reduced. In other words, the algorithm lost the capability of searching through the full 100-D parameter space. For the same reason as in the case of SCE-UA, it is difficult for the PSO algorithm to restore the lost dimensions in the following iterations. For this PSO run, it is lucky that the population degeneration happens because the particles have reached the attractive region of the global minimum and, hence, the subspace contains the global minimum point. However, in all the failed PSO runs, the population degeneration occurs due to the attractive region of a local minimum and the subspace spanned by the population excludes the global minimum. Consequently, the particles lose their capability to approach the global minimum and finally are trapped to the local minimum.

For DE (the panels on the right column), throughout all of the 2500 iterations, every PC remains comparable variance. This means that the searches are always conducted in the 100-D full search space, which contributes to the robust performances of DE in the experiments. In fact, if the degeneration had happened, the way of DE to generate the offspring would not be able to get it recovered too, like in the cases of the other two algorithms.

D. Geometrical Study of the Experiment Results

It is also of great interest to show how the relationship between the global minimum and the subspace spanned by the population changes during the course of evolution for all three algorithms. To do this, we must define some numerical thresholds. The first one is how to define a dimension being reduced numerically. Since the dimensionality of the experiment is 100, for random generated populations, the expected percentage of total variance projected on each dimension is 1%. Therefore, if one dimension has variance projection less than 10^{-2} of its expectation (0.01% of the total variance), it is numerically defined as reduced. Second, it is important to define if the global minimum is within a subspace? In this experiment, if the distance

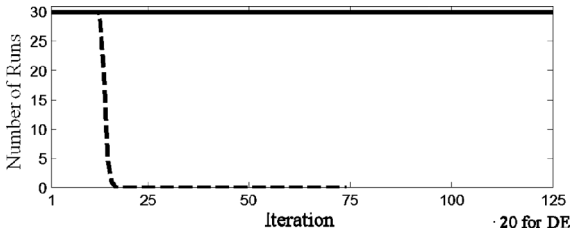


Fig. 6. Number of runs in which the global minimum remains in the space spanned by particle population as a function of iteration through optimization of 100-D CF1 function for SCE-UA(dash line) and DE (solid line) for the ensemble of 30 runs. (Iterations of different algorithm may have different amounts of function evaluation.).

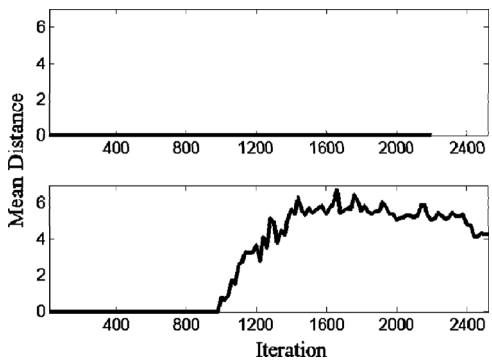


Fig. 7. Mean distance of the global minimum to the hyperplane defined by the particle populations for the 21 successful PSO runs (the upper panel) and 9 failed PSO runs (the lower panel) on the 100-D CF1 function.

from the global minimum to the subspace, which is also a hyperplane, is less than 10^{-6}, is treated as being on the super plane, namely within the subspace.

If we put results of the 30 runs in parallel, after each iteration, we can determine in how many runs (out of the 30 runs) the global minimum is remaining in the subspace spanned by the particle population. Illustrated in Fig. 6, in all SCE-UA runs, the global minimum is excluded from the population space after a few early iterations. In contrast, for all DE runs, the global minimum always remains in the subspace of the population which is actually because the population does not degenerate, as revealed by Fig. 5. From Fig. 5, it is clear that PSO runs all have population degeneration occurring. However, Fig. 7 shows that if the population degeneration is caused by the attractive region of the global minimum ($\hat{g}$), $\hat{g}$ is still contained in the subspace of the population. However, if the degeneration is caused by the attractive region of the local minimum, $\hat{g}$ is very likely to be excluded from the population subspace and when it happens, the search is doomed to fail.

E. Interpretation

Population degeneration cannot be addressed by the population diversity measures, that are proposed by many recent studies in evolutionary algorithms. In general, these reported measures fall into two categories: 1) Population diversity in the objective space [23]–[26]; and 2) Population diversity in the parameter space [27]–[30]. Furthermore, the measures in the parameter space are based on the particle-dimension-distance to quantify population diversity. These distance-based measures cannot identify population degeneration. For instance, in another study [21], we demonstrate that one of the most popular diversity measures cannot reflect the population degeneration that leads the PSO algorithm to fail on some complex test functions.

The difference in the severity of population degeneration between the three algorithms is closely related to the offspring generating mechanisms used by the algorithms. To keep the population spanning the full parameter space through the optimization, offspring should be generated in "all directions" during evolution. Here, "all directions" mean m linearly independent directions in an m-D search space.

In the SCE-UA algorithm, offspring are generated (using the Nelder–Mead simplex scheme) along the steepest descent direction of a benchmark function. The steepest descent direction represents the most promising searching direction, which makes the search effective. Meanwhile the population converges and possesses orientations in accordance to the function features. The complex shuffling scheme and mutation procedure alter the search directions but are not strong enough to prevent the convergence trend in the high-dimensional space. Therefore, the SCE-UA search during the evolution is in "very selective directions," not in all directions.

The scheme employed by PSO generates the offspring in a direction with three components: the direction of the last displacement, the direction toward the best position of the population, and the direction toward the particle's best historical position. The first and last components help maintaining the spatial diversity since each particle's path is independent at the beginning. However, if the similarity among particles' paths emerges, for example the whole population moves from one region to another more promising region during evolution, the spatial diversity of search direction will also be compromised.

In contrast, the DE algorithm generates offspring along directions that are defined by two randomly selected particles and diverted by the hybridization process. These mechanisms tend to generate offspring in all directions and greatly reduce the possibility of population degeneration but cost inefficient searches.

IV. Remedy for Population Degeneration

Because efficiency is essential in high-dimensional optimization, the parsimonious slope-approximation processes of generating offspring, such as the simplex scheme in the SCE-UA method, are preferable in developing evolutionary optimization algorithms for high-dimensional applications. However, the potential risk from population degeneration must be properly controlled. In addition to its capability of discovering population degeneration, PCA also can help remedy the adverse effects of population degeneration. PCA can identify all of the reduced PCs, on which the population variance projections are trivial. Then, extra offspring can be generated along these PCs complementing the offspring generated routinely. In this manner, the offspring is generated in all directions and can span the full parameter space.

As a demonstration, we modified the SCE-UA algorithm [31], [32], by applying PCA to the population at the end of each iteration and generating additional offspring along all reduced PCs. This modified SCE-UA algorithm not only performs properly on the 100-D CF1 function with all runs successfully converging to the global minimum, but also demonstrates its efficiency and effectiveness on a suite of very complex compositional functions.

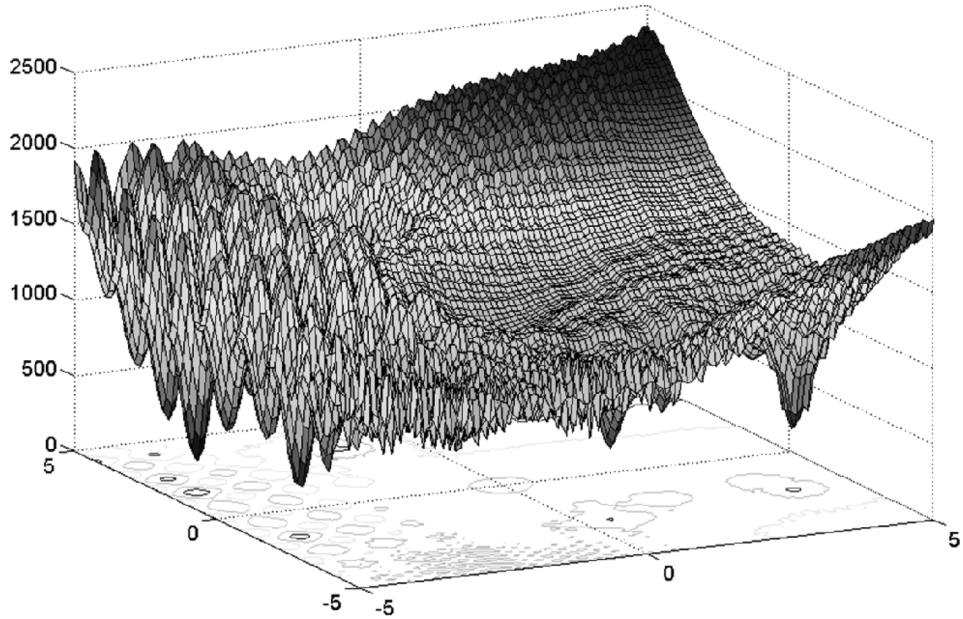


Fig. 8. Response surfaces of benchmark function CF4 in 2-D.

As a general tool, PCA can also be applied to remedy population degeneration associated with PSO. The above experiments show that population degeneration also occurs in the PSO runs on 100-D CF1, but not causing stagnation of the swarm. However, as reported by Chu *et al.* [33], population degeneration triggered by one of the most popular bound-handling schemes, absorbing scheme, does drive the swarm to stagnate at the boundary of the searching space. The absorbing scheme is necessary in exploring the global optima that are close to the boundary in many practical problems. In fact, adding a PCA scheme to PSO, in the same manner as how we apply PCA to SCE-UA, can prevent PSO from stagnation caused by population degeneration when using absorbing bound-handling scheme. This can be demonstrated through a test using the composition function,CF4, in Liang *et al.* [19]. The CF4 function (Fig. 8) constructs a dominant attractive region which converges to a local minimum (at the center of the search space), whereas the global minimum and other better local minima hide in a narrow area close to the boundary.

For comparison, we run PSO 30 times on CF4 with each of the four settings: 1) with reflecting bound-handling: relocating an outside offspring particle inside at the symmetric position by the boundary, 2) with random bound-handling: randomly relocating the outside particles within the search domain, 3) with absorbing bound-handling: setting the outside offspring particles onto the boundary, and 4) with absorbing bound-handling and PCA scheme. Results show that PSO is consistently trapped by the local minimum at the center of the search space (Fig. 9(a)–(c)) with settings 1–3. Only with PCA as in setting 4, PSO can escape the trapping by this local minimum and obtain much better final function values.

V. Practical Application

With remedy of population degeneration, evolutionary algorithms are capable of solving high-dimensional and complex practical problems. As an example, we present some results from our recent study of improving the parameter estimation and calibration of the highly nonlinear hydrologic models used for flood forecasting [34]. In specific, the SACramento Soil-Moisture Accounting Model (SAC-SMA), which is the major component of the U.S. National Weather Service (NWS) River Forecast System and is currently serving as the operational model for flood and river flow forecasting over the United States, was used to examine the SP-UCI method, a SCE-UA variant integrating PCA. The SAC-SMA model has 13 parameters which need to be obtained through parameter estimation.

Simulating the rainfall-runoff process, the SAC-SMA model represents a complex system involving physical components (different soil layers) and water movement (infiltration, percolation). Since many parts of this system are underground and unobservable, it utilizes a series of conceptual water storages to approximate the soil moisture conditions and to control the production of streamflow (Fig. 10). Therefore, the skill of this model relies on how well the model parameters are calibrated.

To demonstrate effectiveness of SP-UCI, we applied it to the calibration of SAC-SMA over the Leaf River basin located in the State of Mississippi. This watershed has an area of 1944 km^2 (Fig. 11) and is an intensively studied watershed, and therefore has abundant and easily accessible hydrological data. We obtained 11 years (January 1, 1953 to December 31, 1963) of observation time series from the Hydrologic Research Laboratory at NWS. The data set includes mean areal precipitation (mm/6 h), potential evapotranspiration (mm/day), and streamflow (m3/s). The mean annual precipitation for the entire period is 1323.7 mm and the mean runoff is 27.14 m3/s.

For comparison, both SP-UCI and SCE-UA were applied 50 independent times, with objective function defined as the daily root-mean-square error (DRMS) of the simulated runoff against the observation. Final DRMS values are presented in Fig. 12, along with results reported by some previous studies [35], [36]. It is evident that SP-UCI elevates SAC-SMA to a record high level in terms of its ability to predict the daily runoff of the basin. SP-UCI not only retrieves the optimal parameter values

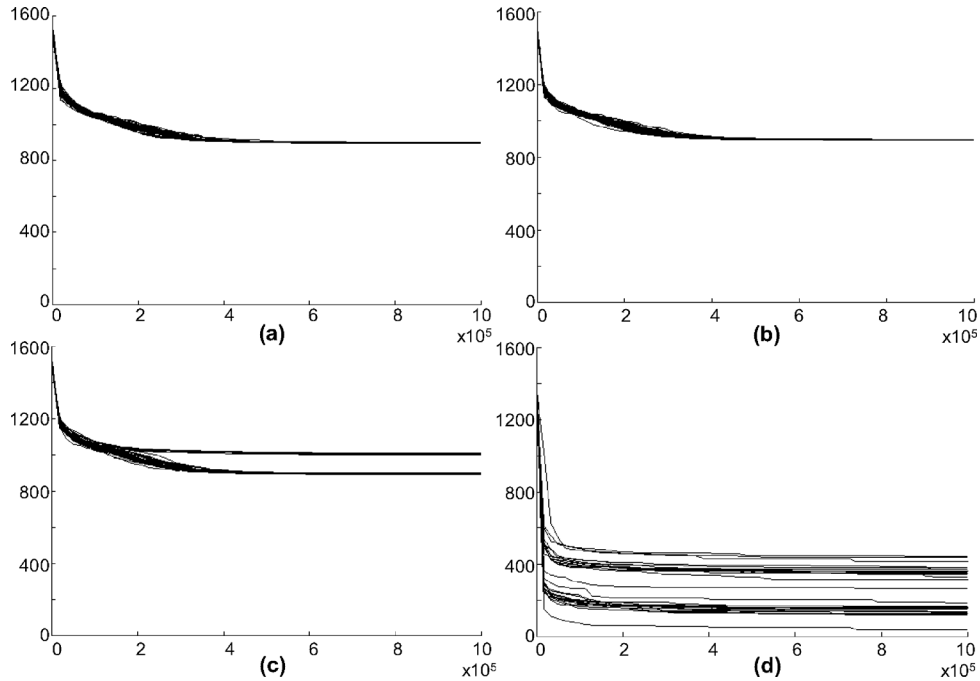


Fig. 9. Fitness curves 30 runs of PSO on 100-D function CF4 with: (a) reflecting bound-handling scheme, (b) random bound-handling scheme, (c) absorbing bound-handling scheme, and (d) absorbing bound-handling scheme and PCA.

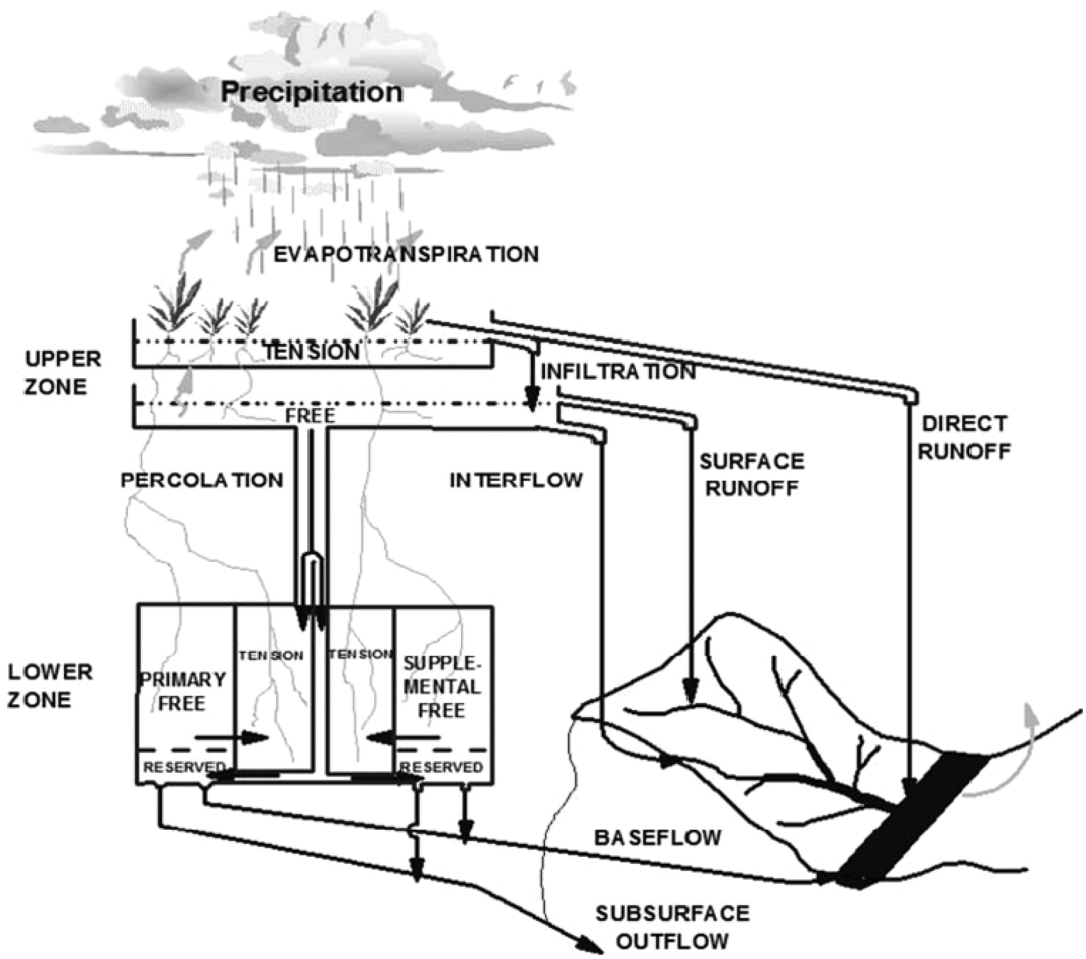


Fig. 10. Schematic of the SAC-SMA model [35].

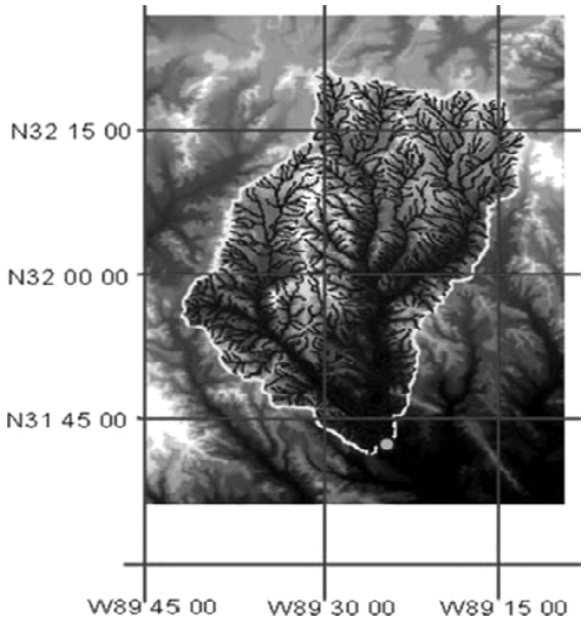


Fig. 11. Study area-the Leaf River basin, Mississippi.

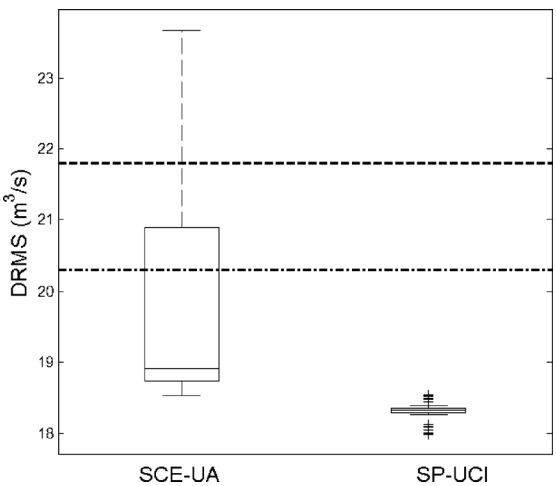


Fig. 12. Box plots of results from 50 runs of SCE-UA and SP-UCI. The dash line and dotted line indicate the result reported by Thiemann *et al.* [36] and Brazil [35] respectively. (The box has lines at the lower quartile, median, and upper quartile values. The whiskers are lines extending from each end of the box to show the extent of the rest of the data. Outliers, denoted by +, are data with values beyond 100 units of interquartile range.).

responsible for more accurate runoff simulation, but also provides consistent (narrow ranges) model parameter distributions, which leads to correct understanding of the model's behavior and the watershed's hydrologic features [34].

VI. Summary and Discussions

In this study, we compare three evolutionary algorithms using two typical benchmark functions, each of which is tested in 30-D and 100-D cases and each test included 30 runs. From the experiments we conclude the following.

1) All the failed global optimizations are unanimously in company with population degeneration in which the population degenerated into a subspace embedded inside the search space and the global minimum is outside of the subspace. This population degeneration is in general irrecoverable by the algorithms themselves.
2) The possibility of the occurrence of population degeneration is closely related to the algorithm's mechanism used to generate offspring. The SCE-UA algorithm has the most effective and efficient scheme to discover better offspring among the three algorithms, but it is most vulnerable to population degeneration. The DE algorithm shows robustness in global optimizations by employing diverse search during the evolution, which, in turn, decreases its inefficiency.
3) DE showed its robustness in this study, but the low efficiency diminishes its application to high-dimensional optimization. On the other hand, in all the succeeded runs, SCE-UA consistently exhibits remarkable efficiency compared with the other two algorithms which suggests that it will have great potential for high-dimensional optimization if the population degeneration can be controlled, as reported in [31], [32].

When using evolutionary algorithms to solve real-world problems, the location of the global optimum (if it exists) is generally unknown. As the searching dimensionality increases, the response surface of the test function becomes much more complicated. Particle population gets less capable to explore the high-dimensional landscape and more susceptible to degeneration. Therefore, the results of this study recommend:

1) An algorithm should maintain the dimensionality of the space spanned by particle population during evolution, because even if only one dimension is reduced, the succeeding search will possibly be restricted to the subspace missing the opportunity to achieve the global minimum.
2) The PCA procedure is a powerful tool of not only discerning population degeneration but also identifying reduced PCs. Hence it can be used to remedy population degeneration and diverse the search directions.
3) As a practical instance of the No Free Lunch Theorems [37], efficiency and robustness are often uncongenial. To be able to work on a wide range of real-world problems, an algorithm should enable users to control the balance between efficiency and robustness. How to realize this through designing an efficient and flexible mechanisms of generating offspring will be one of the foci for our future study.

Acknowledgment

The authors would like to thank Dr. P. N. Suganthan for kindly providing codes of the composition benchmark functions and PSO algorithm. The Matlab codes for DE algorithm were downloaded from Dr. R. Storn's website and the authors also would like to express thanks to him.

References

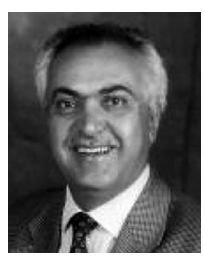
[1] T. Brenner, "Can evolutionary algorithms describe learning processes?," *J. Evol. Econ.*, vol. 8, no. 3, pp. 271–283, 1998.
[2] D. J. Wales and H. A. Scheraga, "Global optimization of clusters, crystals, and biomolecules," *Science*, vol. 285, no. 5432, pp. 1368–1372, 1999.
[3] A. R. Lemmon and M. C. Milinkovitch, "The metapopulation genetic algorithm: An efficient solution for the problem of large phylogeny estimation," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 16, pp. 10516–10521, 2002.

[4] M. Glick, A. Rayan, and A. Goldblum, "A stochastic algorithm for global optimization and for best populations: A test case of side chains in proteins," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 2, pp. 703–708, 2002.
[5] M. A. Nowak and K. Sigmund, "Evolutionary dynamics of biological games," *Science*, vol. 303, no. 5659, pp. 793–799, 2004.
[6] C. Cherniak, Z. Mokhtarzada, R. Rodriguez-Esteban, and K. Changizi, "Global optimization of cerebral cortex layout," *Proc. Nat. Acad. Sci. USA*, vol. 101, no. 4, pp. 1081–1086, 2004.
[7] C. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Trans. Syst., Man, Cybern. B: Cybern.*, vol. 34, no. 2, pp. 997–1006, 2004.
[8] J. Park, K. Lee, J. Shin, and K. Y. Lee, "A particle swarm optimization for economic dispatch with nonsmooth cost functions," *IEEE Trans. Power Syst.*, vol. 20, no. 1, pp. 34–42, 2005.
[9] J. Robinson and Y. Rahmat-Samii, "Particle swarm optimization in electromagnetics," *IEEE Trans. Antennas Propagat.*, vol. 52, no. 2, pp. 397–407, 2004.
[10] M. P. Wachowiak, R. Smolikova, Y. Zheng, J. M. Zurada, and A. S. Elmaghraby, "An approach to multimodal biomedical image registration utilizing particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 289–301, 2004.
[11] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philos. Mag.*, vol. 2, no. 6, pp. 559–572, 1901.
[12] Q. Duan, V. Gupta, and S. Sorooshian, "Shuffled complex evolution approach for effective and efficient global minimization," *J. Optim. Theory Applicat.*, vol. 76, no. 3, pp. 501–521, Mar. 1993.
[13] Q. Duan, S. Sorooshian, and V. Gupta, "Effective and efficient global optimization for conceptual rainfall-runoff models," *Water Resources Res.*, vol. 28, no. 4, pp. 1015–1031, Apr. 1992.
[14] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Computer J.*, vol. 7, no. 4, pp. 308–313, Jan. 1965.
[15] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, "Convergence properties of the Nelder–Mead simplex method in low dimensions," *SIAM J. Optim.*, vol. 9, no. 1, pp. 112–147, Dec. 1998.
[16] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netwarks*, 1995, pp. 1942–1948.
[17] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micromachine Human Sci.*, Nagoya, Japan, 1995, pp. 39–43.
[18] R. Storn and K. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.
[19] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, J, Jun. 2006.
[20] J. Vesterstroem and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Proc. Congr. Evol. Comput.*, 2004, vol. 2, pp. 1080–1987.
[21] W. Chu, X. Gao, and S. Sorooshian, "Bound handling strategy is critical to the success of particle swarm optimization on high-dimensional complex problems," *Inf. Sci.*, no. DOI:10.1016/j.ins.2010.11. 030, 2010.
[22] C. T. Kelley, "Detection and remediation of stagnation in the Nelder–Mead algorithm using a sufficient decrease condition," *SIAM J. Optim.*, vol. 10, no. 1, pp. 43–55, Oct. 1999.
[23] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evol. Comput.*, vol. 10, no. 3, pp. 263–282, 2002.
[24] P. K. Tripathi, S. Bandyopadhyay, and S. K. Pal, "Multi-Objective Particle Swarm Optimization with time variant inertia and acceleration coefficients," *Inf. Sci.*, vol. 177, no. 22, pp. 5033–5049, 2007.
[25] S. Aine, R. Kumar, and P. P. Chakrabarti, "Adaptive parameter control of evolutionary algorithms to improve quality-time trade-off," *Appl. Soft Comput.*, vol. 9, no. 2, pp. 527–540, 2009.
[26] X. Shen, M. Zhang, and T. Li, "A multi-objective optimization evolutionary algorithm addressing diversity maintenance," in *Int. Joint Conf. Computational Sciences and Optimization*, Apr. 2009, pp. 524–527.
[27] J. Riget and J. S. Vesterstroem, A Diversity-Guided Particle Swarm Optimizer–The ARPSO Univ. Aarhus, Dept. Comput. Sci., Aarhus, Denmark, 2002, EVA Life, Technical Report 2002–02.
[28] M. Lozano, F. Herrera, and J. R. Cano, "Replacement strategies to preserve useful diversity in steady-state genetic algorithms," *Inf. Sci.*, vol. 178, no. 23, pp. 4421–4433, 2008.
[29] Q. Kang, L. Wang, and Q. Wu, "A novel ecological particle swarm optimization algorithm and its population dynamics analysis," *Appl. Math. Comput.*, vol. 205, no. 1, pp. 61–72, 2008.
[30] Y. Zhao, W. Zu, and H. Zeng, "A modified particle swarm optimization via particle visual modeling analysis," *Comput. Math. With Applicat.*, vol. 57, no. 11–12, pp. 2022–2029, 2009.
[31] W. Chu, "Evolutionary Optimization Methods for High-Dimensional Complex Systems: Theory, Algorithm, and Application to Rainfall-Runoff Models," Ph.D., Univ. California, Dept. Civil Environmental Eng., Irvine, CA, 2009.
[32] W. Chu, X. Gao, and S. Sorooshian, "A new evolutionary search strategy for global optimization of high-dimensional problems," *Inf. Sci.*, to be published.
[33] W. Chu, X. Gao, and S. Sorooshian, "Bound handling strategy is critical to the success of particle swarm optimization on high-dimensional complex problems," *Inf. Sci.*, no. Doi:10.1016/j.ins.2010.11.030.
[34] W. Chu, X. Gao, and S. Sorooshian, "Improving the shuffled complex evolution scheme for optimization of complex nonlinear hydrological systems: Application to the calibration of the Sacramento soil-moisture accounting model," *Water Resources Res.*, no. DOI: 10.1029/2010WR009224.
[35] L. E. Brazil, "Multilevel Calibration Strategy for Complex Hydrologic Simulation Models," Ph.D. dissertation, Colorado State Univ., Fort Collins, 1988.
[36] M. Thiemann, M. Trosset, H. Gupta, and S. Sorooshian, "Bayesian recursive parameter estimation for hydrologic models," *Water Resources Res.*, vol. 37, no. 10, pp. 2521–2535, 2001.
[37] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

Wei Chu received the B.S. degree from Nanjing University, Nanjing, China, the M.S. degree from Stanford University, Stanford, CA, and the Ph.D. degree from the University of California at Irvine. His research is focused on evolutionary optimization of high-dimensional complex problems with strong interest in solving real world problems.

Xiaogang Gao received the Ph.D. degree from the Department of Hydrology and Water Resources, University of Arizona, Tucson, in 1993.

Since 2003, he has been a Professor in the Department of Civil and Environmental Engineering, University of California, Irvine. His research interests include hydrology and water resource management, coupled atmosphere-land surface modeling, optimization, and statistical data assimilation.



Soroosh Sorooshian received his Ph.D. degree in engineering from University of California at Los Angeles, in 1978.

He is a member of the US National Academy of Engineering (NAE) and is currently a Distinguished Professor at University of California at Irvine. His areas of interest include system simulation and analysis, operations research, decision analysis, hydroclimate modeling, and stochastic parameter estimation.